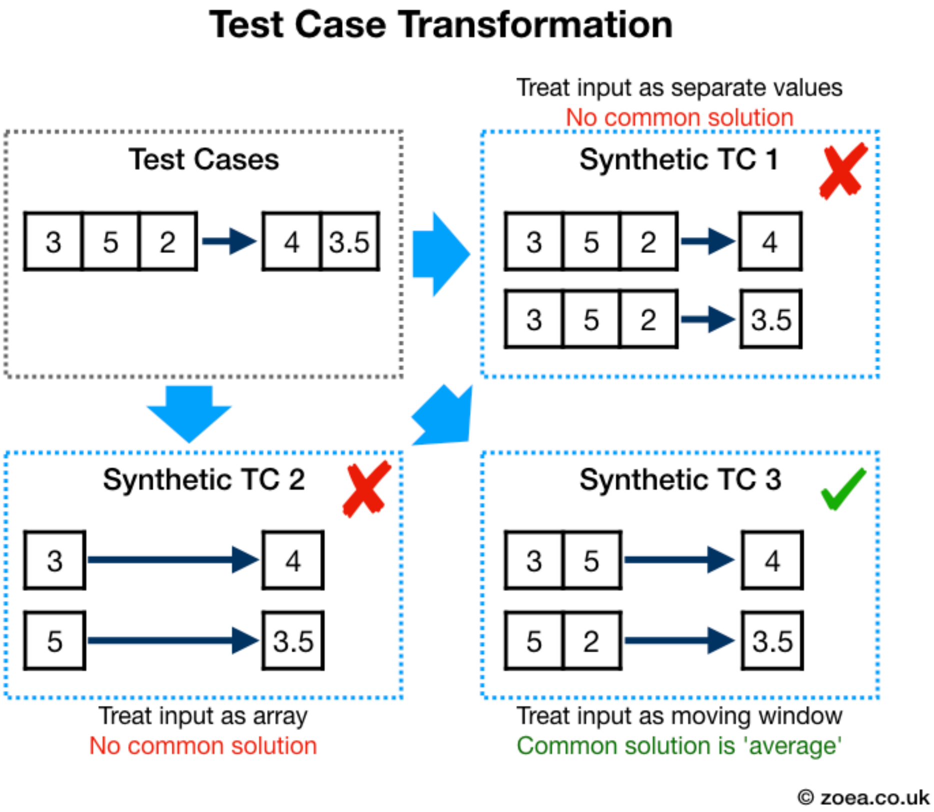
Zoea

Research note 24

Synthetic Test Cases

Edward McDaid & Sarah McDaid
18 Jan 2023

Zoea uses synthetic test cases to represent internal data transformations. What are they and why are they so useful?



Every Zoea program regardless of whether it is written in Zoea Specification Language or drawn as a Zoea Visual diagram corresponds to a set of test cases. A test case is simply an example of a program input and the corresponding output. The inputs and outputs can be any sort of data including numbers and strings. These can also be combined into composite data structures such as arrays and hashmaps. The user can define any number of test cases but - unless the program is intended to do nothing - there is normally at least one case.

Internally Zoea transforms the set of test cases for a program into a large number of other sets of test cases that it creates itself. We call these synthetic test cases to reflect the fact that they are not created by the user. The transformations that create the synthetic test cases are not random but rather pre-defined. Each transformation is carried out by a kind of expert that specialises in one particular aspect of programming language or software development knowledge. These experts are called knowledge sources. Each synthetic test case represents a different hypothesis about a fragment of the required code.

For example, one of the knowledge sources that specialises in arrays will trigger if it sees an array in the input and an array of the same length in the output. It will then create a new set of synthetic test cases where the input is each separate element of the input array and the output is the corresponding single element of the output array. This hypothesis corresponds to a code fragment that iterates over all of the elements in turn - possibly with some as yet unspecified transformation. This particular knowledge source does not care what, if any, further transformations may be required to produce the rest of the solution. Similarly, it has no interest in how the data for the synthetic test case was produced. Its job is done if it can add its own specialist piece of knowledge as a suggestion. Everything else is up to other knowledge sources to figure out.

There are many different knowledge sources in Zoea and these often create many different synthetic test cases. Knowledge sources simply ignore any test cases that are not applicable to them. Also, no two sets of synthetic test cases that were derived from the same test cases are identical. This prevents endless cycles of synthetic test cases corresponding to code that adds no value.

Each set of synthetic test cases that is created is treated as a completely new problem. This means that it is processed in exactly the same way as the original test case. Synthetic test cases are considered by all knowledge sources but because the synthetic test cases are different from the original test cases it is likely that a different set of experts will be activated. This will involve the creation of many further synthetic test cases, and so on. The whole process stops when the input for a synthetic test case is the same as the output - in which case a solution has been found. Alternatively the process can time out so that it does not continue forever.

Synthetic test cases provide a separate context for every different part of a complex problem solving process. This serves to isolate the associated state information, simplifies the management of large quantities of data and makes retrieval easier.

The combination of a set of synthetic test cases and a knowledge source is called an activation. It can also be thought of as the allocation of a knowledge source to a set of synthetic test cases. Activations are used in scheduling to create cluster jobs so that the work of executing the knowledge source for a particular synthetic test case can be carried out by a different CPU core or worker node. The scheduler is relatively simple but this is all of the coordination that is required for Zoea to work.

Each synthetic test case has a record of its parent activation as well as its own code hypothesis. When a solution is found all of the code fragments for the activations that connect the original test case to the solution are combined. This is how the complete code for the solution is produced.

In Zoea there is no privileged software component that makes decisions or looks out for a solution. Instead, these concerns are delegated to all of the knowledge sources and the scope of any decisions applies only to the individual concerned. This can be viewed as self-organising behaviour.

The synthetic test case approach is simple and consistent. Treating intermediate results in the same way as input makes the design and development of Zoea easier and more reliable. In particular knowledge sources are easy to develop because they all have the same standard interface and they can be tested in isolation before being integrated. Also, there are no dependencies between any of the components. Each one has a specific job and the synthetic test case is all of the information that is needed to accomplish it. It is also an easy matter to inspect what is happening to the data at each step during processing.

Different programs will need very different combinations of knowledge sources in order to produce a solution. These will also have to be applied in different orders and combined to form diverse and potentially complex structures. It would therefore be virtually impossible to spell out the precise logic required to produce any possible program. In Zoea the order in which knowledge sources are applied is not specified but rather determined automatically at runtime for each given problem.

The use of synthetic test cases emerged early on in the development of Zoea. This was driven by the strict enforcement of principles that Zoea should be as simple and consistent as possible. In retrospect it would have been much more difficult had we tried to build Zoea without them.

Learn more at **zoea.co.uk**